|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Question | Answer | Expected Answer | Result |
| 001 | Can you tell me the number of movies in the inventory grouped by category? | SELECT category.name AS category\_name, COUNT(film.film\_id) AS film\_count FROM film JOIN film\_category ON film.film\_id = film\_category.film\_id JOIN category ON film\_category.category\_id = category.category\_id GROUP BY category\_name; | N/A | Works on PSQL and produces output. Need to confirm. |
| 002 | Create a query that lists each film category and the number of films made in that category. | WITH total\_category AS (SELECT film\_category.film\_id, category.name FROM film\_category JOIN category ON film\_category.category\_id = category.category\_id) SELECT tc.name, COUNT(tc.film\_id) AS num\_films FROM total\_category tc GROUP BY tc.name; | SELECT DISTINCT category\_name,  COUNT(film\_title) OVER(PARTITION BY category\_name) AS category\_count  FROM  (SELECT f.title film\_title, c.name category\_name  FROM film f  JOIN film\_category fc ON fc.film\_id = f.film\_id  JOIN category c ON c.category\_id = fc.category\_id) t1  ORDER BY category\_count; | Works on PSQL and produces output. Need to confirm. |
| 003 | Create a query that lists each movie, the film category it is classified in, and the number of times it has been rented out. | SELECT f.film\_id, c.category\_name, COUNT(r.rental\_id) AS number\_of\_times FROM film f JOIN film\_category cf ON f.film\_id = cf.film\_id AND cf.category\_id = c.category\_id JOIN rental r ON r.inventory\_id = f.film\_id GROUP BY f.film\_id, c.category\_name ORDER BY number\_of\_times DESC NULLS LAST; | SELECT DISTINCT film\_title, category\_name,  COUNT(rental\_id) OVER(PARTITION BY film\_title) AS rental\_count  FROM (SELECT f.title film\_title, c.name category\_name, r.rental\_id rental\_id  FROM film f  JOIN film\_category fc ON fc.film\_id = f.film\_id  JOIN category c ON c.category\_id = fc.category\_id  JOIN inventory i ON i.film\_id = f.film\_id  JOIN rental r ON r.inventory\_id = i.inventory\_id) t1  ORDER BY category\_name,film\_title; | Incorrect SQL syntax – Table c missing. Failed. |
| 004 | Create a query that lists each actor's full name, film release year and number of films released per year. | SELECT a.actor\_full\_name AS "Actor Name", to\_char(f.release\_date,'YYYY') AS "Release Year", COUNT(\*) filter (WHERE c.category\_id = 7) AS "Number of Films" FROM actor a JOIN film f ON a.actor\_id = f.film\_id JOIN film\_category c ON f.film\_id = c.film\_id GROUP BY a.actor\_full\_name, to\_char(f.release\_date,'YYYY'); | SELECT CONCAT(a.first\_name,' ',a.last\_name) actor\_name,  COUNT(f.release\_year) year\_wise\_film  FROM actor a  JOIN film\_actor fa ON fa.actor\_id = a.actor\_id  JOIN film f ON f.film\_id = fa.film\_id  GROUP BY actor\_name  ORDER BY COUNT(f.release\_year); | Incorrect SQL syntax – actor\_full\_name does not exist. Failed.  Incorrect question?? |
| 005 | Create a query that lists each actor's full name and number of films released per year. | SELECT c.city\_name, COUNT(film.release\_year) AS num\_films FROM customer c JOIN film f ON c.customer\_id = f.customer\_id GROUP BY c.city\_name; | SELECT CONCAT(a.first\_name,' ',a.last\_name) actor\_name,  COUNT(f.release\_year) year\_wise\_film  FROM actor a  JOIN film\_actor fa ON fa.actor\_id = a.actor\_id  JOIN film f ON f.film\_id = fa.film\_id  GROUP BY actor\_name  ORDER BY COUNT(f.release\_year); | Incorrect SQL, syntax error |
| 006 | Create a list of each actor and number of films released per year in which they have acted | WITH film\_actor AS (SELECT actor\_id, film\_id FROM film\_actor ), category\_film AS (SELECT film\_id, category\_id FROM film\_category), inventory\_inventory AS (SELECT inventory\_id, store\_id FROM inventory) SELECT a."first\_name" || ' ' || a."last\_name" AS actor, to\_char(f.release\_date,'YYYY') AS year, COUNT(\*) filter (WHERE category\_id IS NOT NULL AND category = 1070) AS comedy, COUNT(\*) filter (WHERE category\_id IS NOT NULL AND category = 1077) AS drama, COUNT(\*) filter (WHERE category\_id IS NOT NULL AND category = 1080) AS action FROM actor a JOIN film\_actor f ON a.actor\_id = f.actor\_id LEFT JOIN category\_film g ON g.film\_id = f.film\_id AND g.category\_id IS NOT NULL WHERE g.category = 1070 GROUP BY actor, year UNION SELECT a."first\_name" || ' ' || a."last\_name" AS actor, to\_char(f.release\_date,'YYYY') AS year, COUNT(\*) filter (WHERE category\_id IS NOT NULL AND category = 1077) AS drama FROM actor a JOIN film\_actor f ON a.actor\_id = f.actor\_id LEFT JOIN category\_film g ON g.film\_id = f.film\_id AND g.category\_id IS NOT NULL WHERE g.category = 1077 GROUP BY actor, year UNION SELECT a."first\_name" || ' ' || a."last\_name" AS actor, to\_char(f.release\_date,'YYYY') AS year, COUNT(\*) filter (WHERE category\_id IS NOT NULL AND category = 1080) AS action FROM actor a JOIN film\_actor f ON a.actor\_id = f.actor\_id LEFT JOIN category\_film g ON g.film\_id = f.film\_id AND g.category\_id IS NOT NULL WHERE g.category = 1080 GROUP BY actor, year ORDER BY actor, year NULLS LAST; | SELECT CONCAT(a.first\_name,' ',a.last\_name) actor\_name,  COUNT(f.release\_year) year\_wise\_film  FROM actor a  JOIN film\_actor fa ON fa.actor\_id = a.actor\_id  JOIN film f ON f.film\_id = fa.film\_id  GROUP BY actor\_name  ORDER BY COUNT(f.release\_year); | Incorrect SQL |
| 007 | Create a query that lists each customer's full name, total payment he has done and categorizes the customer as per payment into Bronze, Silver, Gold and Platinum classes | WITH customer\_payment AS (SELECT c.customer\_id, c.first\_name, c.last\_name, p.payment\_date, p.amount  FROM customer c JOIN payment p ON c.customer\_id = p.customer\_id), total\_payments  AS (SELECT customer\_id, SUM(amount) AS total\_payment FROM customer\_payment  GROUP BY customer\_id), vip\_class AS (SELECT tp.customer\_id,  CASE WHEN tp.total\_payment > 10000 THEN 'Platinum'  WHEN tp.total\_payment BETWEEN 500 AND 1000 THEN 'Gold' ELSE 'Silver' END AS vip\_class  FROM total\_payments tp) SELECT cp.first\_name, cp.last\_name, tp.total\_payment, vc.vip\_class  FROM customer\_payment cp JOIN total\_payments tp ON cp.customer\_id = tp.customer\_id  JOIN vip\_class vc ON tp.customer\_id = vc.customer\_id;  SELECT COUNT(\*) AS total\_customers FROM customer; | SELECT CONCAT(c.first\_name,' ',c.last\_name) customer\_full\_name,  SUM(p.amount) as total\_amount,  CASE  WHEN SUM(p.amount)<50 THEN 'Bronze Customer'  WHEN SUM(p.amount)>50 and SUM(p.amount)<=100 THEN 'Silver Customer'  WHEN SUM(p.amount)>100 and SUM(p.amount)<=150 THEN 'Gold Customer'  ELSE 'Platinum Customer'  END as amount\_section  FROM payment p  JOIN customer c ON c.customer\_id = p.customer\_id  GROUP BY 1  ORDER BY SUM(p.amount); | Works on PSQL and produces output. 2 queries created. Need to confirm |
|  | List the names of each actor in the database alongwith a count of movies in which they have acted | SELECT a.first\_name, a.last\_name, COUNT(f.film\_id) AS film\_count FROM actor a JOIN film\_actor fa ON a.actor\_id = fa.actor\_id JOIN film f ON fa.film\_id = f.film\_id GROUP BY a.first\_name, a.last\_name ORDER BY film\_count DESC NULLS LAST |  | Correct response and SQL |
|  | List the titles of each movie that has been rented out atleast two times | SELECT film.film\_title FROM rental JOIN film ON rental.inventory\_id = film.film\_id GROUP BY film.film\_title HAVING COUNT(rental\_id) >= 2 ORDER BY film.film\_title NULLS LAST |  | Error in query – invalid column name: film\_title instead of title |